KMS: atomic modeset/pageflip

Rob Clark

What is it?

- Atomic pageflip
  - Updating CRTC fb and/or one or more plane fb's atomically (in a single vblank)
  - Also possibly adjusting properties: z-order, alpha blending modes, rotation, colorspace-conversion coefficients, etc
  - 'test' flag to allow userspace to check a proposed configuration first

- Atomic modeset
  - Configuring one or more CRTCs
  - 'test' flag to allow checking if the proposed combination of timings/resolutions are supported by the hw

Why do we need it?

- Atomic pageflip
  - Compositors using overlay planes to bypass GPU for compositing surfaces
    - Need to keep bypassed surface state (size, position, fb) in sync w/ GPU composition output on CRTC layer
    - Need to know when they'll hit hw limits about overlay plane sizes/scaling/etc
      - Some limits may be with combinations of multiple enabled planes
      - So not easy to express limits statically to userspace
- Atomic modeset
  - Userspace needs to know valid combinations of settings for multi-display
  - Memory bandwidth limits, etc, may mean that certain resolutions are possible with single display but not multiple displays

Property-ification..

- The proposed solution configures *everything* via properties
- We need to support taking a list of properties anyways
- Doing everything via properties means:
  - common code-paths
  - Future extensibility
- But then how does error checking work?
  - Ie. valid fb dimensions, position, etc
  - Short version: it is still there, but moves from the ioctl handler fxn
  - Long version: on next slides

Splitting mode object mutable state

- What is in 'struct drm_{crtc,plane,etc}' is combination of:
  - Mode state set from userspace: fb, {src,crtc}_{x,y,w,h}, etc
  - Other: possible_crtcs, list head, funcs, etc
- For 'test' steps, we need to build up proposed state, and rollback
  - Split into 'struct drm_{crtc,plane,etc}_state' simplifies things
    - Just a single pointer to update to commit changes
    - We could probably simplify crtc helpers change rollback this way
  - Split out of state structs also lets us use helpers to:
    - Avoid a lot of property nonsense in each driver for common properties
    - Re-introduce the standard error checking lost from ioctl handler

```
int drm_plane_check_state(struct drm_plane *plane, struct drm_plane_state *state);
void drm_plane_commit_state(struct drm_plane *plane, struct drm_plane_state *state);
int drm_plane_set_property(struct drm_plane *plane, struct drm_plane_state *state,
                 struct drm_property *property, uint64_t value);
```

- (And same for CRTC and eventually connector)

Splitting mode object mutable state (cont)

- Also, property values array moved into state structs
  - Automatically keeps userspace visible property values in sync
  - Don't get property values confused by 'test' step or failed config changes

```
struct drm_plane_state {
     struct drm_crtc *crtc;
     struct drm_framebuffer *fb;

    /* Signed dest location allows it to be partially off screen */
    int32_t crtc_x, crtc_y;
    uint32_t crtc_w, crtc_h;

    /* Source values are 16.16 fixed point */
    uint32_t src_x, src_y;
    uint32_t src_h, src_w;

    struct drm_object_property_values propvals;
};
```

- Drivers should wrap state structs w/ their own to add driver specifics:

```
struct omap_plane_state {
    struct drm_plane_state base;
    uint8_t rotation;
    uint8_t zorder;
};
```

Atomic funcs

- atomic_begin(dev) - allocate state token
- atomic_check(dev, state) – check proposed state
  - Use drm_*_check_state() for common stuff
- atomic_commit(dev, state) – commit proposed state
  - Do driver specific stuff, then drm_*_commit_state()
- atomic_end(dev, state) – cleanup/deallocate
- Example:

```
state = dev->driver->atomic_begin(dev);

 if (page_flip->flags & DRM_MODE_PAGE_FLIP_EVENT)
    e = create_vblank_event(dev, file_priv, page_flip->user_data);

 for (i = 0; i < page_flip->count_props; i++)
    drm_mode_set_obj_prop_id(dev, state,
                prop.obj_id, prop.obj_type,
                prop.prop_id, prop.value);

ret = dev->driver->atomic_check(dev, state);

 if (!(page_flip->flags & DRM_MODE_TEST_ONLY))
    ret = dev->driver->atomic_commit(dev, state, e);

dev->driver->atomic_end(dev, state);
```

Other misc changes

- Object property type
  - DRM_MODE_PROP_OBJECT
  - To set crtc, fb, etc as a property
- Dynamic property flag
  - DRM_MODE_PROP_DYNAMIC
  - Hint to userspace about properties which can be safely changed without 'test' step
- Signed property ranges
  - DRM_MODE_PROP_SIGNED
  - Uses signed integer comparison to check for valid property values

TODO

- Don't remove plane->update_plane(), crtc->page_flip() yet
  - These are no longer needed for 'property-ified' drivers
  - But probably better to have a transition period, than port all drivers at once

- Still tweaking ioctl struct

Questions?